AMENDMENTS TO THE CLAIMS

Please cancel claims 2, 4, 12, 14, and 22, without prejudice or disclaimer of the subject matter; and amend claims 1, 5 to 8, 11, 15 to 18, and 21, as shown below. This listing of claims replaces all prior versions and listings of claims in the application:

Listing of Claims:

1. (Currently Amended) A method for allocating memory in a computer system, the method comprising:

determining a size of a memory page used by a paged virtual memory system;

outputting a request from an application to ~~an operating~~ the page virtual memory system for allocation of a block of memory by ~~the operating~~ an operating system to the application, the block of memory being integer *N* times the size of the memory page;

accessing the block of memory for the application;

dividing the block of memory into ~~a plurality of~~ (*N*-1) frames, with each of the ~~plurality of~~ frames operable to store an indexing structure associated with an attribute of a data record, and each frame being the same size as the memory page used by the operating system;

determining a beginning page boundary of a first whole memory page within the block of memory;

storing the frames beginning at the beginning page boundary;

dividing each of the frames into a plurality of instances, with each of the plurality of instances operable to store an index node of the indexing structure, the index node including left and right pointers pointing to other index nodes of the index structure having the same attribute as the index node;

storing administrative data in a cut-off portion of the block of memory disposed in front of the beginning page boundary or behind the (*N*-1)*th* frame; and

associating the attribute with the plurality of instances for data storage using the plurality of instances.

2. (Cancelled)

3. (Original) The method of claim 1 wherein the size of the block of memory is determined by a coding parameter associated with the application.

4. (Canceled)

5. (Currently Amended) The method of ~~claim 4~~ claim 1 wherein ~~dividing the block of memory into frames further includes designating a portion of the block of memory before the first page boundary as unused~~ storing administrative data includes storing administrative data in the cut-off portion disposed behind the (*N*-1)*th* frame and the method further comprises designating the cut-off portion in the front of the beginning page as unused.

6. (Currently Amended) The method of claim 1 wherein a size of each ~~frame~~ of the frames is determined by a coding parameter.

7. (Currently Amended) The method of claim 1 wherein a size of each ~~frame~~ of the frames is determined by a page size used by the operating system.

8. (Currently Amended) The method of claim 1 wherein ~~dividing a block of memory into frames includes:~~ storing administrative data includes storing administrative data in the cut-off portion disposed in front of the beginning page boundary and the method further comprises designating the cut-off portion disposed behind the (*N*-1)*th* frame as unused.

~~determining a last page boundary within the block of memory; and~~

~~designating a portion of the block of memory after the last page boundary as unused.~~

9. (Original) The method of claim 1 wherein a single type of data is stored in the block of memory.

10. (Original) The method of claim 1 wherein data from a fast query system is stored in the instances.

11. (Currently Amended) A software application tangibly embodied on a computer-readable medium using application-level memory management, the software application comprising:

an application-level memory manager operable to:

determine a size of a memory page used by the paged virtual memory system,

request from the page virtual memory system to allocate a block of memory to store data elements, the block of memory being integer *N* times the size of the memory page,

~~to~~ divide the block of memory into ~~a plurality of~~ (*N*-1) frames, with each of the ~~plurality of~~ frames operable to store an indexing structure associated with an attribute of a data record, and each of the frames being the same size as the memory page used by the operating system,

determine a beginning page boundary of a first whole memory page within the block of memory,

store the frames beginning at the beginning page boundary,

~~and further to~~ divide each frame into a plurality of instances, with each of the plurality of instances operable to store an index node of the indexing ~~structure;~~ structure, the index node including left and right pointers pointing to other index nodes of the index structure having the same attribute as the index node, and

store administrative data in a cut-off portion of the block of memory disposed in front of the beginning page boundary or behind the (*N*-1)*th* frame; and

application code operable to associate the attribute with the plurality of instances for storage of the data elements in the plurality of instances.

12. (Cancelled)

13. (Original) The software application of claim 11 wherein the size of the block of memory is determined by a coding parameter.

14. (Cancelled)

15. (Currently Amended) The software application of ~~claim 14~~ claim 11 wherein ~~a portion of the block of memory before the first page boundary is designated as unused~~ to store administrative data the application-level memory manager is operable to store administrative data in the cut-off portion disposed behind the (*N*-1)*th* frame and the application-level memory manager is further operable to designate the cut-off portion in the front of the beginning page as unused.

16. (Currently Amended) The software application of claim 11 wherein a size of each ~~frame~~ of the frames is determined by a coding parameter.

17. (Currently Amended) The software application of claim 11 wherein a size of each ~~frame~~ of the frames is determined by the page size used by the operating system.

18. (Currently Amended) The software application of claim 11 wherein ~~the block of memory includes a last page boundary and a portion of the block of memory after the last page boundary is designated as unused~~ to store administrative data the application-level memory manager is operable to store administrative data in the cut-off portion disposed in front of the beginning page boundary and the application-level memory manager is further operable to designate the cut-off portion disposed behind the (*N*-1)*th* frame as unused.

19. (Original) The software application of claim 11 wherein a single type of data is stored in the block of memory.

20. (Original) The software application of claim 11 wherein the application code implements a fast query system.

21. (Currently Amended) The method of claim 1 further comprising:

~~associating data elements used by an application with an application-defined instance type;~~

~~associating the application-determined instance type with an application-determined one of a plurality of blocks of memory allocated by an operating system, wherein the application-determined memory block is divided into a plurality of frames, with each of the plurality of frames operable to store an indexing structure associated with an attribute of a data record, the plurality of frames being further divided into a plurality of instances, with each of the plurality of instances operable to store an index node of the indexing structure; and~~

populating the plurality of instances with the data elements.

22. (Cancelled)

23. (Original) The method of claim 21 further comprising:

removing the data elements; and

returning the block of memory to the operating system.

24. (Original) The method of claim 23 wherein returning the block of memory to the operating system comprises:

returning the block of memory to a buffer; and

determining after a pre-determined period of time that the block of memory is no longer required by the application.